Feuille d'exercices n°1 – numération

1) **Conversion de la base b vers la base 10**

a) *De la base 2 vers la base 10*
Convertir en base 10 les nombres suivants :

$$A = 101001_{(2)} \qquad B = 10110011_{(2)} \qquad C = 1100101_{(2)} \qquad D = 100010111_{(2)}$$

b) *De la base 7 vers la base 10*
Convertir en base 10 les nombres suivants :

$$E = 36_{(7)} \qquad F = 435_{(7)} \qquad G = 6610_{(7)}$$

c) *De la base 16 vers la base 10* Convertir en base 10 les nombres suivants :

$$H = 81A_{(16)} \qquad I = 20BF3_{(16)} \qquad J = C0039_{(16)} \qquad K = ABCD_{(16)} \qquad L = E3F5_{(16)}$$

2) **Conversion de la base 10 vers la base b**

Dans ces exercices, on utilisera les deux méthodes exposées dans le cours, et on en comparera l'efficacité.

a) *De la base 10 vers la base 2*
Donner l'écriture en base 2 des nombres suivants :

$$M = 19_{(10)} \qquad N = 31_{(10)} \qquad O = 256_{(10)} \qquad P = 729_{(10)}$$

b) *De la base 10 vers la base 3*
Donner l'écriture en base 3 des nombres suivants :

$$Q = 18_{(10)} \qquad R = 76_{(10)} \qquad S = 729_{(10)}$$

c) *De la base 10 vers la base 16* Donner l'écriture en base 16 des nombres suivants :

$$T = 70_{(10)} \qquad U = 471_{(10)} \qquad V = 718_{(10)} \qquad W = 51727_{(10)}$$

3) **Conversion binaire-hexadécimal**

Dans ces exercices, on passera directement d'une base à l'autre *sans passer par la base* 10.

a) *Du binaire vers l'hexadécimal*
Donner l'écriture en base 16 des nombres suivants :

$$X = 101101_{(2)} \qquad Y = 101101011110_{(2)} \qquad Z = 100111001110111_{(2)}$$

b) *De l'hexadécimal vers le binaire*
Donner l'écriture en base 2 des nombres suivants :

$$A' = 24D_{(16)} \qquad B' = 70EC_{(16)} \qquad C' = 8BA_{(16)} \qquad D' = EF36_{(10)}$$

4) **Opérations en binaire**

Dans cet exercice, tous les nombres donnés sont en base 2. On calculera directement en binaire, puis on vérifiera le résultat en convertissant en base 10.

a) *addition*
Calculer :

$$E' = 1011 + 101 \qquad F' = 1010101010111001 + 1111011011011110$$

b) *soustraction*
Calculer : $G' = 11001101 - 1001011$.

c) *multiplication ou division par une puissance de* 2
Calculer le produit H' de $11000_{(2)}$ par $8_{(10)}$, puis le quotient I' de $111011101_{(2)}$ par $32_{(10)}$.

d) *multiplication ou division par un nombre quelconque*

- Calculer les produits :

$$J' = 11000 * 11 \qquad K' = 11011101 * 11110011$$

- Calculer le quotient : $L' = 11110100/1101$.

e) **Débordements ?**

- Si l'on effectue le calcul de F' sur un microprocesseur dont les registres ont 16 bits, que se passe-t-il ?
- Pour additionner entre eux deux nombres de 16 bits, un microprocesseur "16 bits" suffit-il ?
- Pour multiplier entre eux deux nombres de 16 bits, de quel genre de microprocesseur a-t-on besoin ?
- À votre avis, comment faisaient les ingénieurs pour faire des calculs sur des "grands" nombres lorsque les microprocesseurs n'avaient que des registres sur 8 bits ?

5) **Bonus : opérations en complément à** 2

a) *conversion*

(i) Comment sont représentés 0, $1_{(10)}$, $-1_{(10)}$ en complément à 2 sur 8 bits ? Dans ce système, quel sont le plus grand et le plus petit nombres qu'on peut coder ?
(ii) Convertir $83_{(10)}$ et $-107_{(10)}$ en complément à 2 sur 8 bits, puis sur 16 bits.
(iii) Quels entiers sont codés 01110110 et 10110110 en complément à 2 sur 8 bits ?
(iv) Quel est le complément à 2 sur 8 bits de -128 ?

b) *addition*
Calculer $1 + (-1)$, puis $-32 + 43$, puis $-43 + 32$ en complément à 2 sur 8 bits, avant de convertir le résultat en décimal.

c) *soustraction*
Comment soustraire deux nombres en complément à 2 ? Calculer $103 - 87$ en passant en binaire.

d) *multiplication*
Voici une méthode pour effectuer le produit de deux nombres codés en complément à 2, extraite de la page Wikipédia anglaise sur le complément à 2 :

> The product of two n-bit numbers requires $2n$ bits to contain all possible values. If the precision of the two two's-complement operands is doubled before the multiplication, direct multiplication (discarding any excess bits beyond that precision) will provide the correct result. For example, take $6 \times (-5) = -30$. First, the precision is extended from 4 bits to 8. Then the numbers are multiplied, discarding the bits beyond 8 (shown by 'x'):

```
   00000110  (6)
 x 11111011  (-5)
 ==========
        110
       110
      000
     110
    110
   110
  x10
 xx0
 ==========
 xx11100010  (-30)
```

Sur ce modèle, calculer le produit de -15 et -14, puis le produit de 132 et -223.